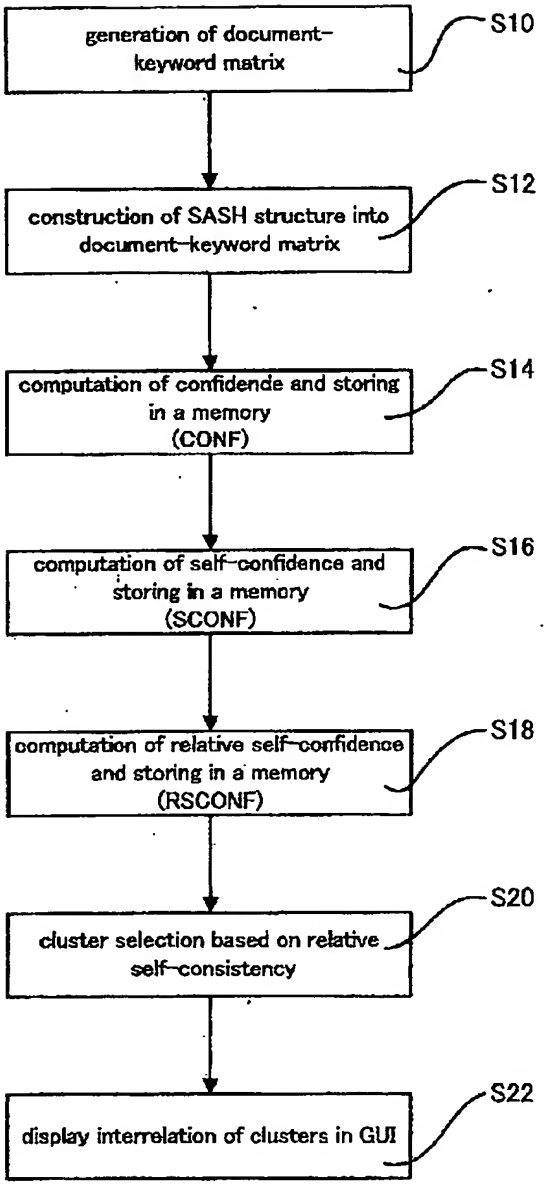

generation of document-keyword matrix
S10
construction of SASH structure into document-keyword matrix
S12
computation of confidende and storing in a memory (CONF)
S14
computation of self-confidence and storing in a memory (SCONF)
S16
computation of relative self-confidence and storing in a memory (RSCONF)
S18
cluster selection based on relative self-consistency
S20
display interrelation of clusters in GUI
S22

FIG 1

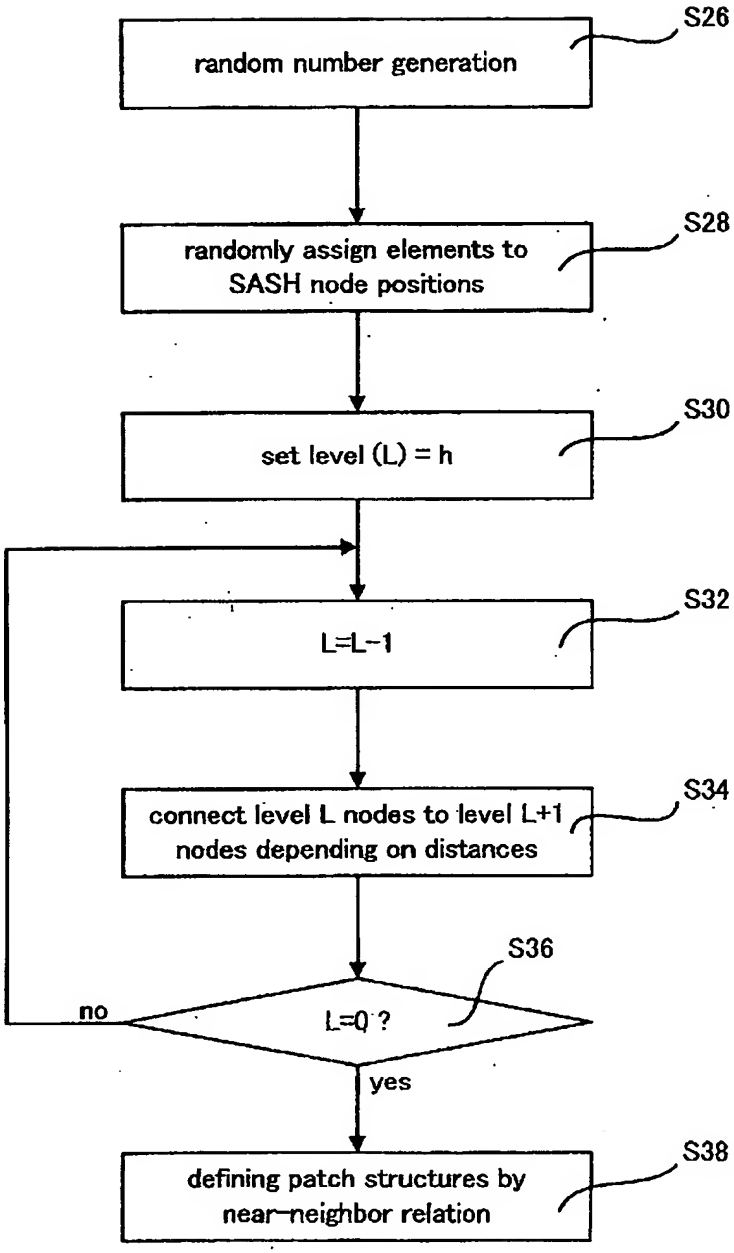

random number generation
S26
randomly assign elements to SASH node positions
S28
set level (L) = h
S30
L=L-1
S32
connect level L nodes to level L+1 nodes depending on distances
S34
L=0 ?
S36
no
yes
defining patch structures by near-neighbor relation
S38

FIG 2

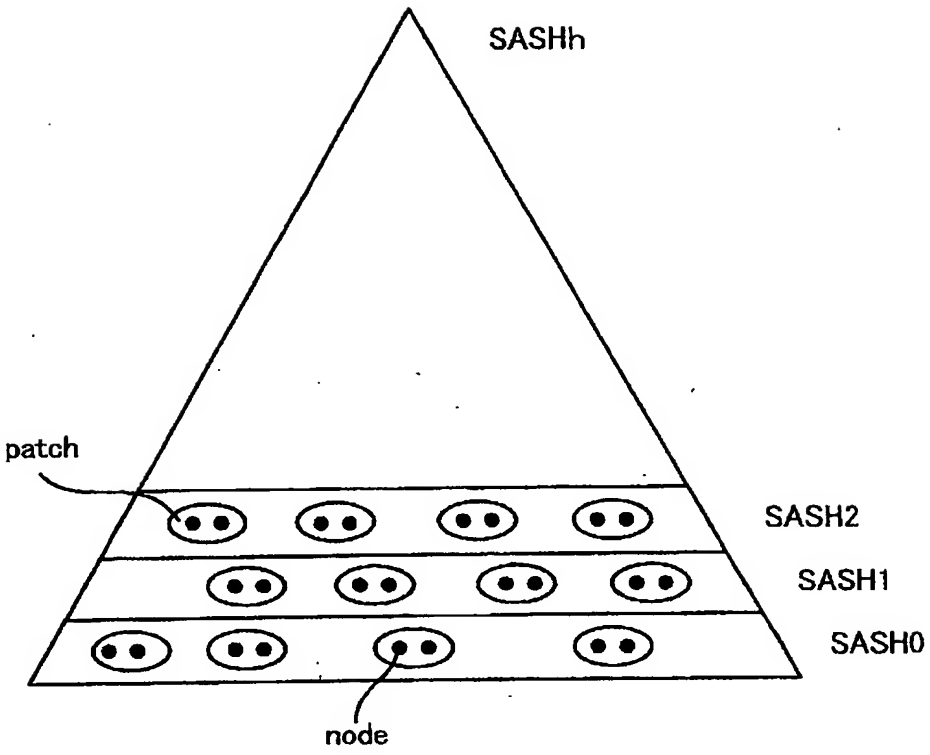


FIG 3

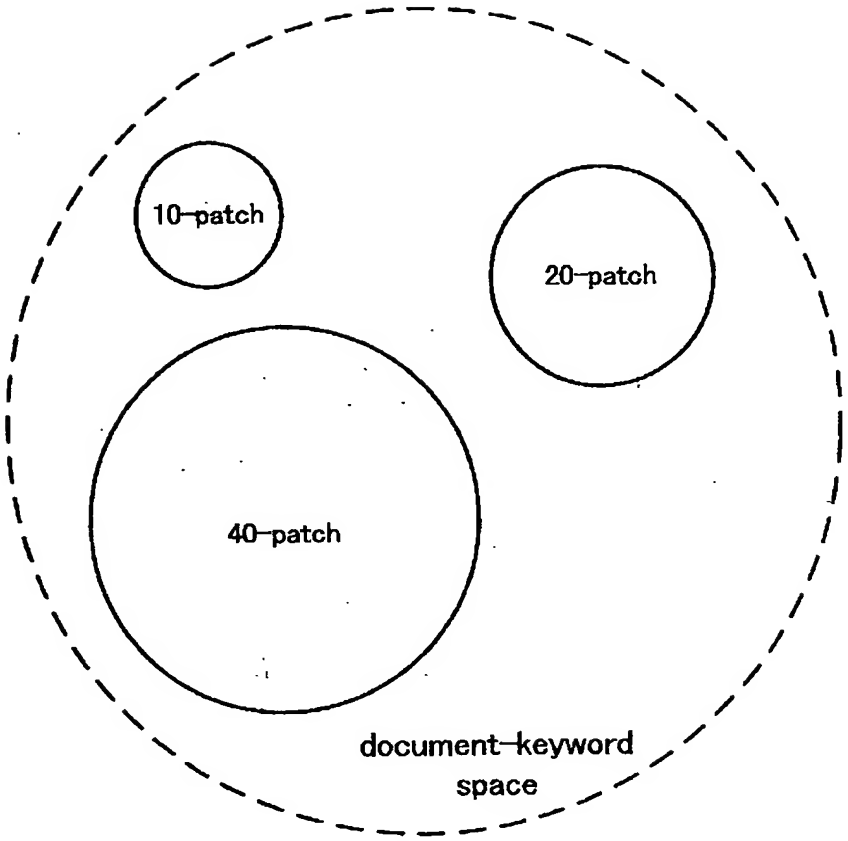
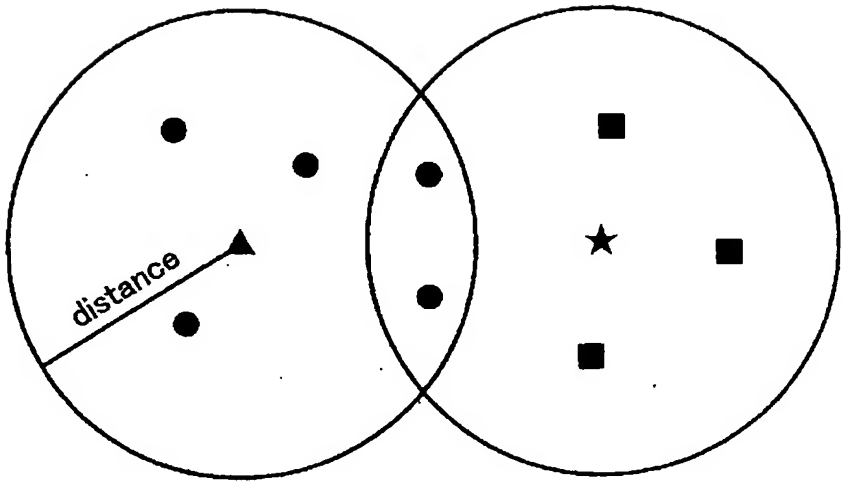


FIG 4



$CONF(C_i, C_j) = 2/5 = 40\%$

FIG 5

```
Profile (query q; maximum patch size m): SCONF list SCONFL
    {Let QNL be the m-patch precomputed for query q.}
    {Let NNL be a list of the m-patches precomputed for every element of QNL.}
    {Initially, w.count = ∅ is assumed for every element v in the data set.}
1.  score ← 0;
    {Initially, no query neighbors are in the current patch.}
    for i = 1 to m do
2.      QNL[i].count ← 0;
    end for
    for i = 1 to m do
        {Retrieve the number of times QNL[i] has been encountered as an external neighbor so far.}
3.      score ← score + QNL[i].count;
        {Indicate that henceforth QNL[i] is in the current i-patch.}
4.      QNL[i].count ← present;
        for j = 1 to i − 1 do
5.          w ← NNL[j, i];
            if w.count = present then
6.                  score ← score + 1;
            else if w.count ≥ 0 then
7.                  w.count ← w.count + 1;
            end if
8.          w ← NNL[i, j];
            if w.count = present then
9.                  score ← score + 1;
            else if w.count ≥ 0 then
10.                 w.count ← w.count + 1;
            end if
        end for
11.     w ← NNL[i, i];
        if w.count = present then
12.         score ← score + 1;
        else if w.count ≥ 0 then
13.         w.count ← w.count + 1;
        end if
14.     SCONFL[i] = score/i²;
    end for
    {Reset the counts to their default value.}
    for i = 1 to m do
15.     QNL[i].count ← ∅;
    end for
```

FIG 6

SASH #	patch list	CONFL	SCONFL	RSCONFL
0	$NN(R_{1'0,0},m)$, •••	•••	•••	•••
1	$NN(R_{1'1,0},m)$, •••	•••	•••	•••
⋮	⋮	⋮	⋮	⋮
h	$NN(R_{1'0,0},m)$, •••	•••	•••	•••

FIG 7

```
RefineProfile (query q;
               inner patch size k_I;
               outer patch size k_O): reordered query k_I-patch RQNL
  {Let QNL be the k_O-patch precomputed for query q.}
  {Let NNL be a list of the k_O-patches precomputed for every element of QNL.}
  {Initially, v.inpatch = false is assumed for every element v in the data set.}
  {Identify the inner patch members.}
  for i = 1 to k_I do
1.    QNL[i].inpatch ← true;
  end for
  {Initialize the confidence value CONFc of every patch element to zero.}
  for i = 1 to k_O do
2.    CONFc[i] ← 0;
  end for
  {For each element of the outer patch, count the number of elements
     of their k-nearest-neighbor sets shared with that of q.}
  for i = 1 to k_O do
     for j = 1 to k_I do
3.      w ← NNL[i, j];
        if w.inpatch = true then
4.          CONFc[i] ← CONFc[i] + 1;
        end if
     end for
5.   CONFc[i] ← CONFc[i]/k_O;
  end for
  {Reorder the outer patch elements according to their confidence values, from highest to lowest.}
6. RQNL ← sort(QNL, CONFc, k_O);
  {Reset the patch membership indicators to their default values.}
  for i = 1 to k_I do
7.    QNL[i].inpatch ← false;
  end for
```

FIG 8

PatchCluster (data set S;
RSCM parameters a, b, $m = \varphi(b)$;
Thresholds $\alpha, \beta, \gamma, \delta$): query cluster graph G

1. Randomly partition the set S into subsets S_t of approximate size $\frac{|S|}{2^t}$, for $0 \leq t \leq h = \lfloor \log_2 |S| \rfloor$.
2. For all $0 \leq t \leq h$ do:
   (a) For every element $v \in S_t$, compute nearest-neighbor patches $\mathbf{NN}(R_t, v, m)$, where $R_t = \bigcup_{i \geq t} S_i$.
   (b) For each element $v_{t,i} \in S_t$, compute the optimal query cluster size $k(v_{t,i})$ maximizing $\mathbf{RSCONF}(\mathbf{NN}(R_t, v_{t,i}, k), \varphi)$, for values of k between a and b
   The ranked collection of patches
   $$\mathcal{C}_t = \langle C_{t,i} | i < j \Rightarrow \mathbf{RSCONF}(C_{t,i}, \varphi) \geq \mathbf{RSCONF}(C_{t,j}, \varphi) \rangle$$
   form the candidates for the query clusters associated with sample $R_t \subseteq S$, where $C_{t,i} = \mathbf{NN}(R_t, v_{t,i}, k(v_{t,i}))$ and $C_{t,j} = \mathbf{NN}(R_t, v_{t,j}, k(v_{t,j}))$.
   (c) Let Q_t be a list of patches of $\mathcal{C}_t$ that have been confirmed as query clusters of R_t. Initially, Q_t is empty.
   (d) For all $1 \leq i \leq |\mathcal{C}_t|$ do:
      i. If $\mathbf{RSCONF}(C_{t,i}, \varphi) < \alpha$, then break from the loop.
      ii. For all $w \in C_{t,i}$ do: if $\mathbf{NN}(R_t, w, k) \notin |\mathcal{C}_t|$ for any value of k, or failing that, if $\max\{\mathbf{CONF}(\mathbf{NN}(R_t, w, k), C_{t,t}), \mathbf{CONF}(C_{t,i}, \mathbf{NN}(R_t, w, k))\} < \beta$, then add $C_{t,i}$ to the list Q_t.
3. Let h' be the largest index for which $|Q_{h'}| > 0$. Let $\{C_{t,j}\}$ be the set of patches comprising Q_t, where $C_{t,j} = \mathbf{NN}(R_t, q_{t,j}, k(q_{t,j}))\}$, for all $0 \leq t \leq h'$. Initialize the node set of the query cluster graph G to contain these patches, one patch per node.
4. For all $\delta \leq t \leq h'$, all $1 \leq j \leq |Q_t|$, and all $\max\{0, t-\delta\} \leq s \leq t$, do:
   (a) Compute $C'_{t,j} = \mathbf{NN}(R_s, q_{t,j}, 2^{t-s} k(q_{t,j}))\}$.
   (b) For all $1 \leq i \leq |Q_s|$, if $C_{s,i} \neq C'_{t,j}$ and $\max\{\mathbf{CONF}(C_{s,i}, C'_{t,j}), \mathbf{CONF}(C'_{t,j}, C_{s,i})\} \geq \gamma$, then introduce the edges $(C_{s,i}, C_{t,j})$ and $(C_{t,j}, C_{s,i})$ into G, with weights $\mathbf{CONF}(C_{s,i}, C'_{t,j})$ and $\mathbf{CONF}(C'_{t,j}, C_{s,i})$, respectively.

FIG 9

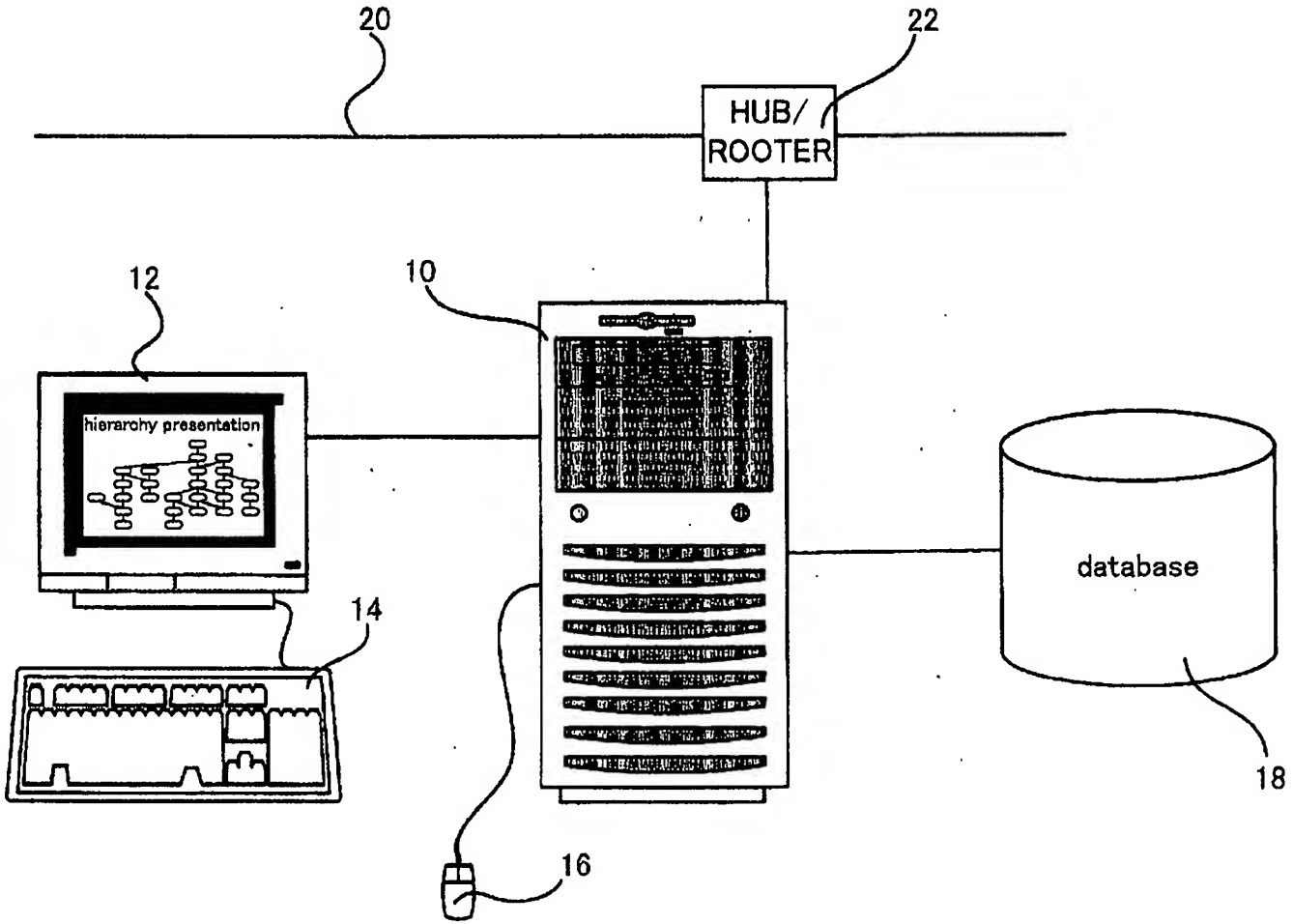


FIG 10

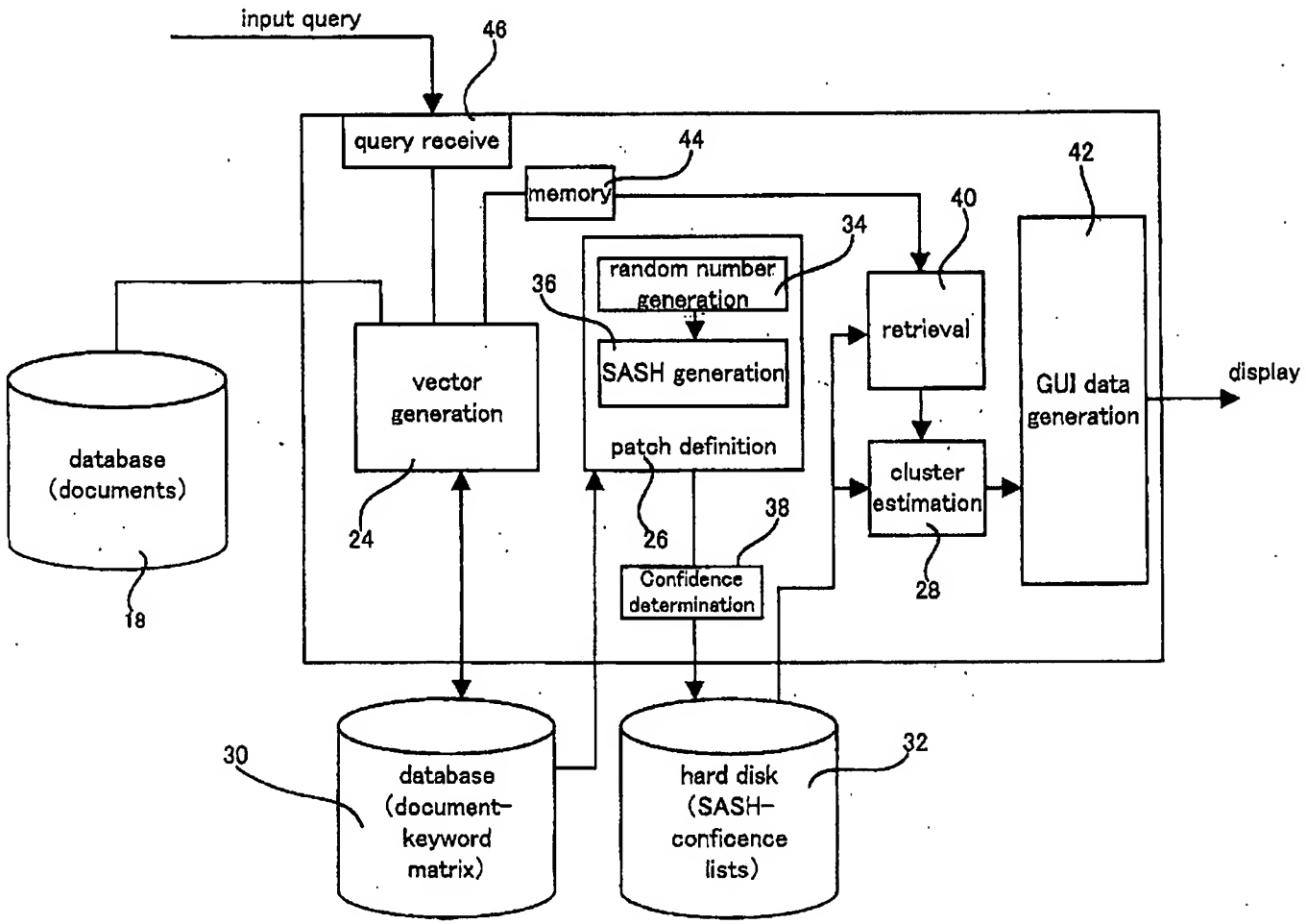


FIG 11

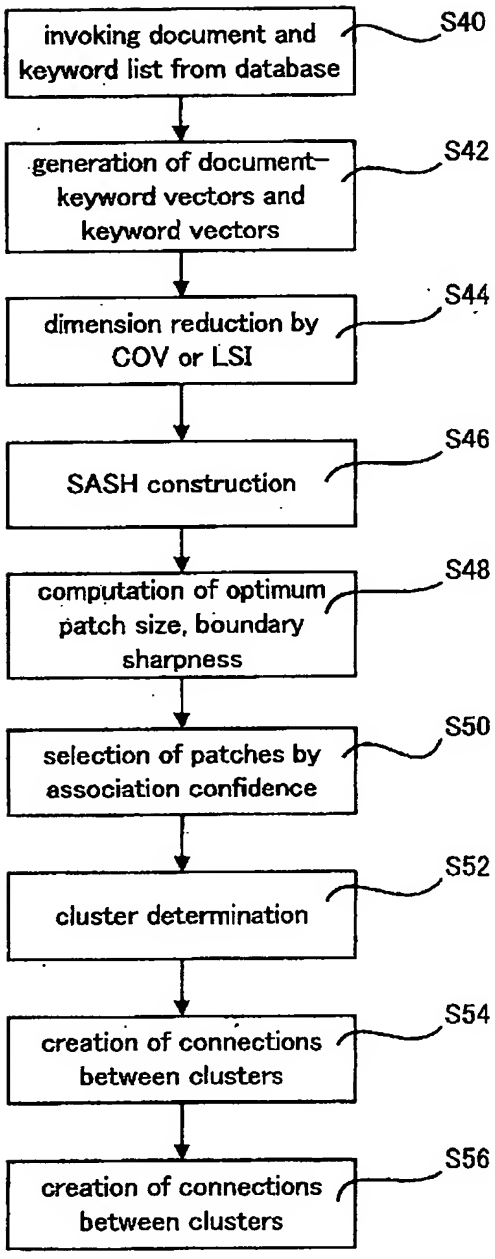

invoking document and keyword list from database
S40
generation of document-keyword vectors and keyword vectors
S42
dimension reduction by COV or LSI
S44
SASH construction
S46
computation of optimum patch size, boundary sharpness
S48
selection of patches by association confidence
S50
cluster determination
S52
creation of connections between clusters
S54
creation of connections between clusters
S56

FIG 12

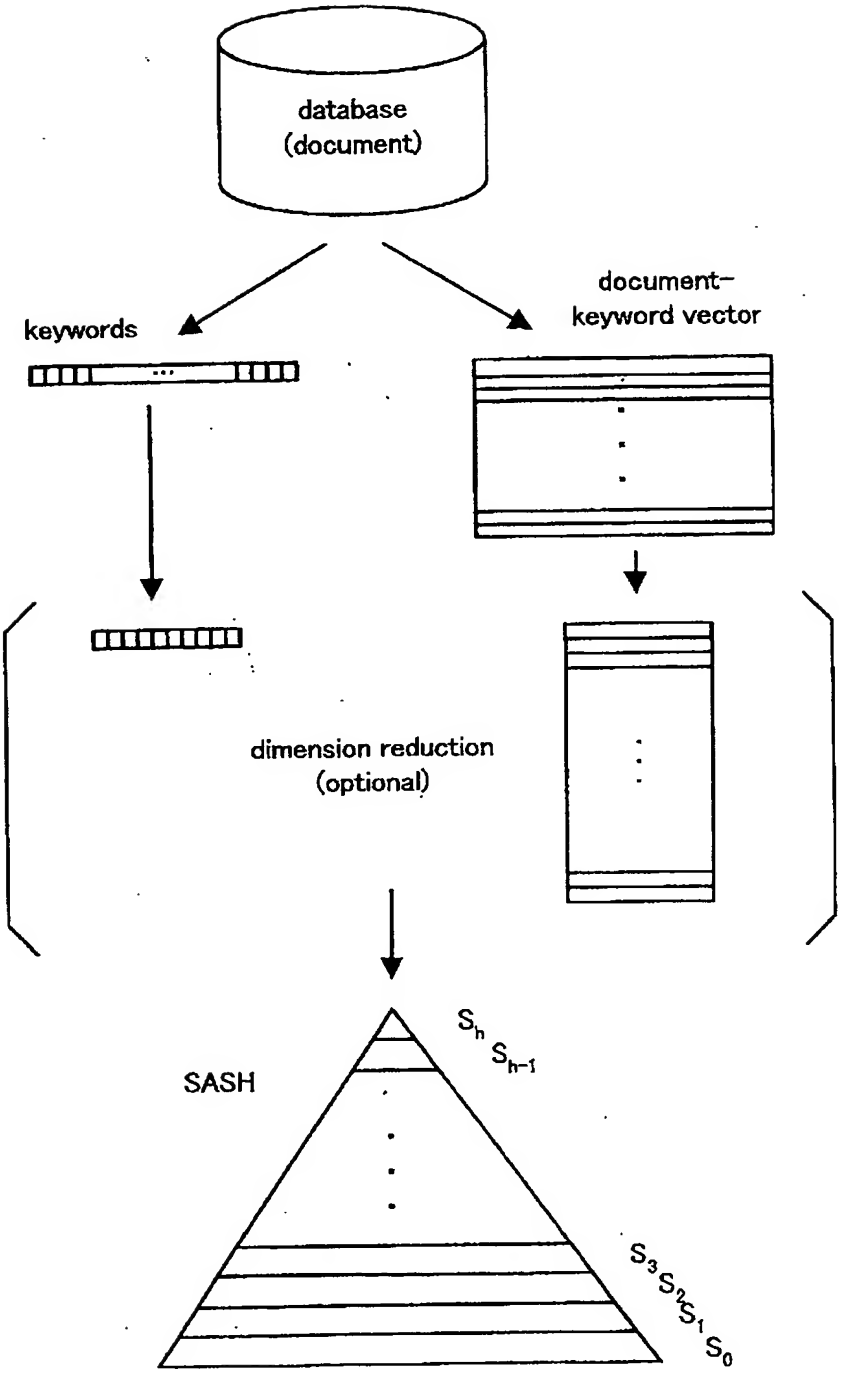
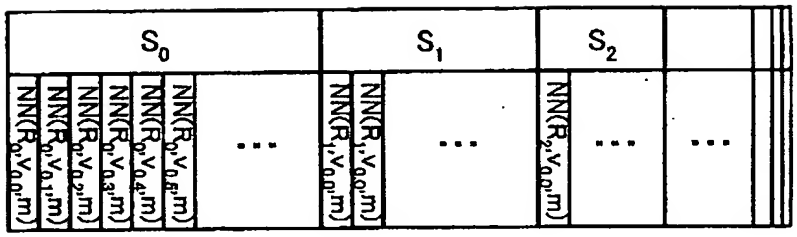
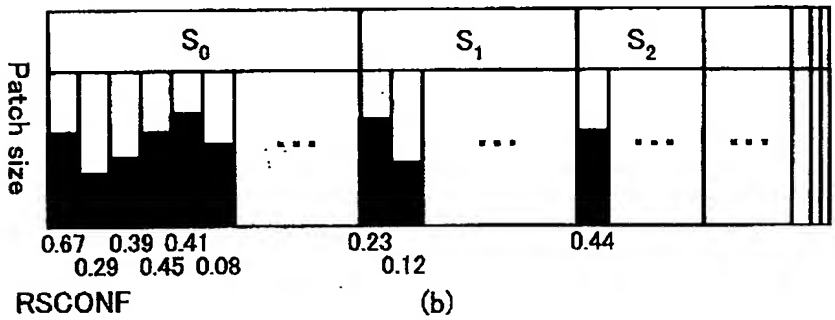


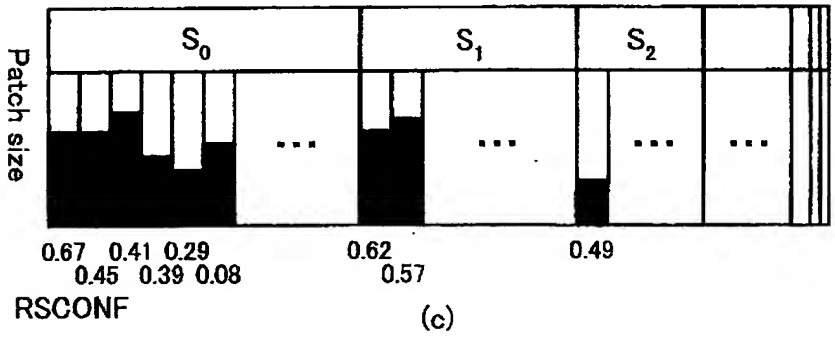
FIG13



(a)

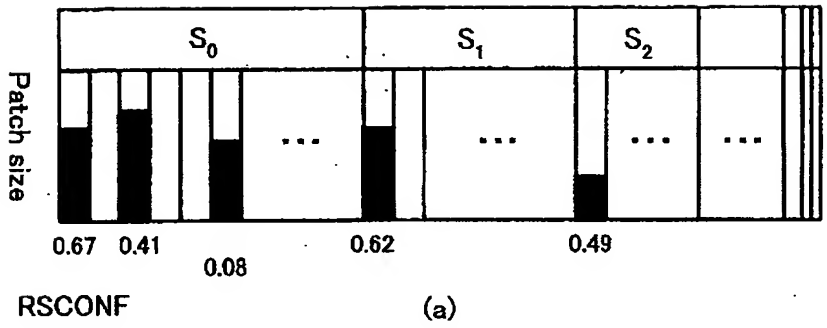


(b)



(c)

FIG 14



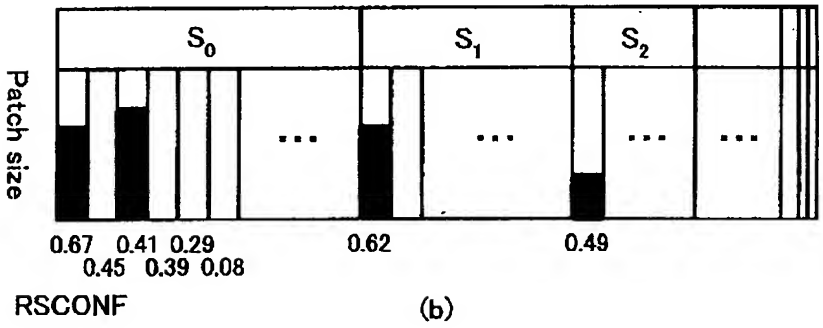


FIG 15

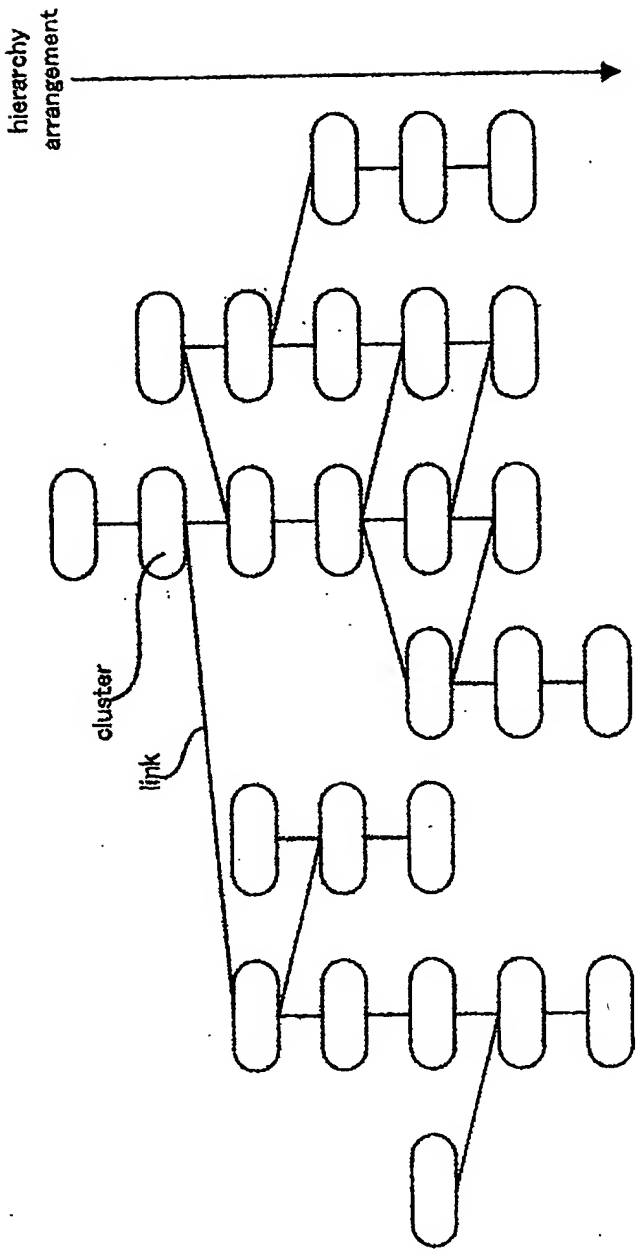


FIG 16

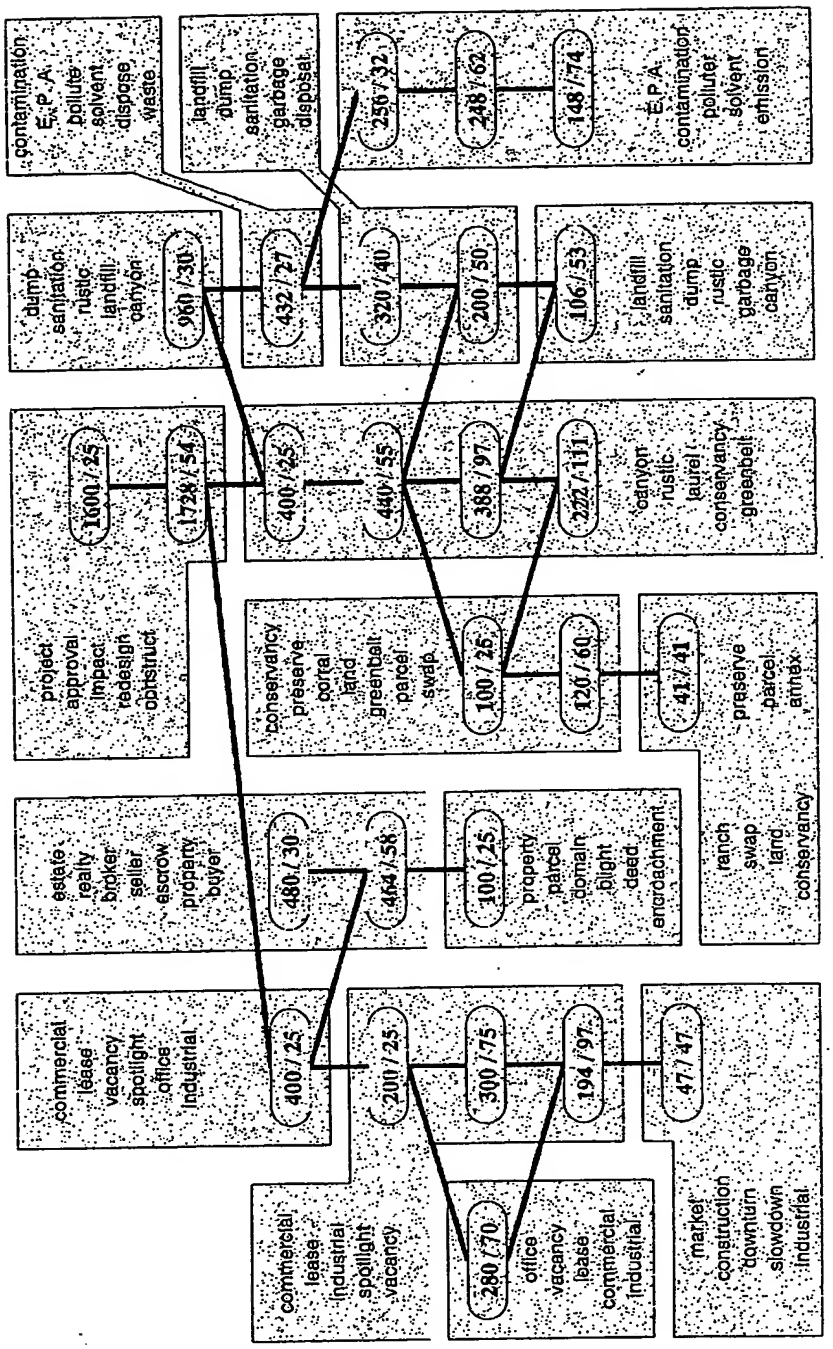


FIG 17

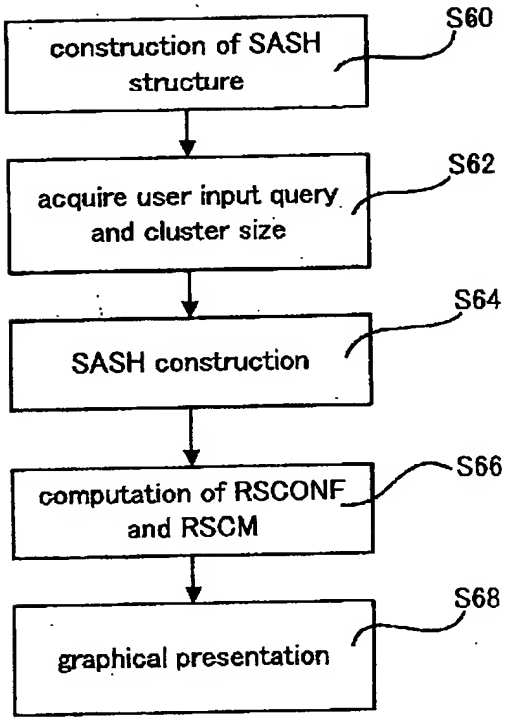


FIG 18

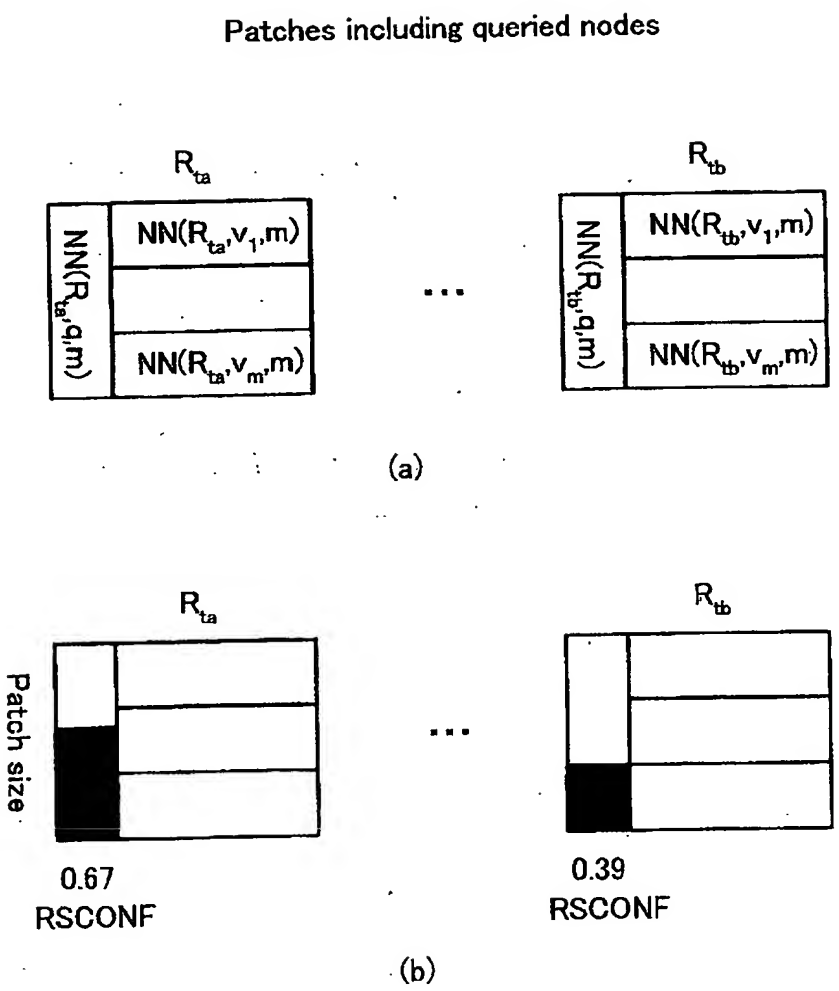


FIG 19

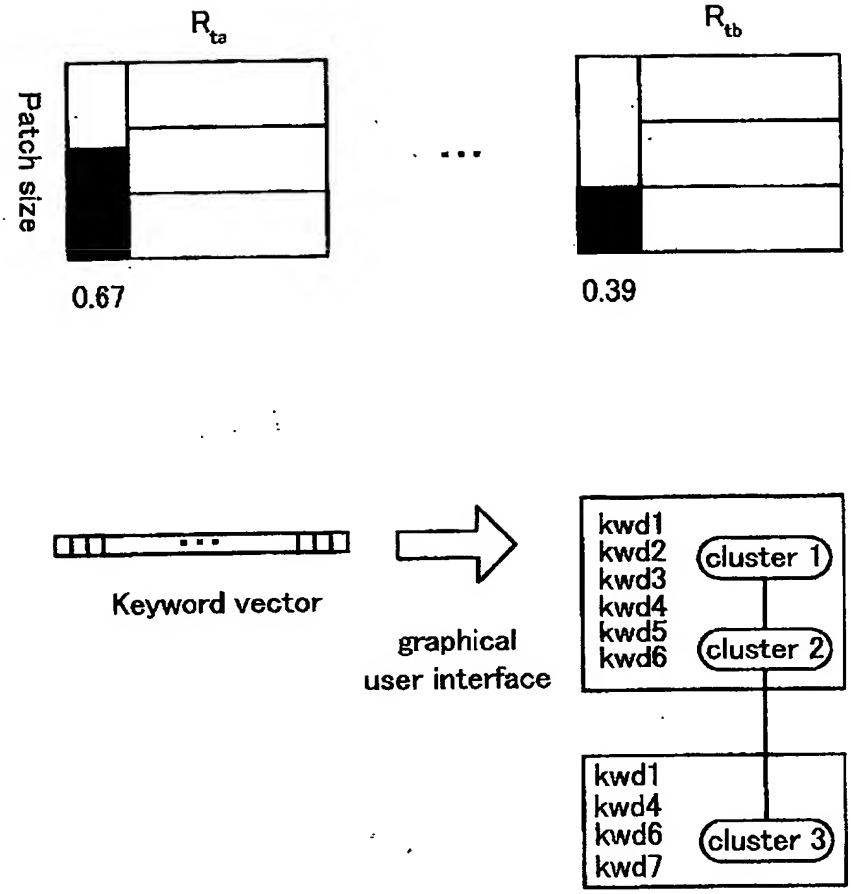


FIG 20

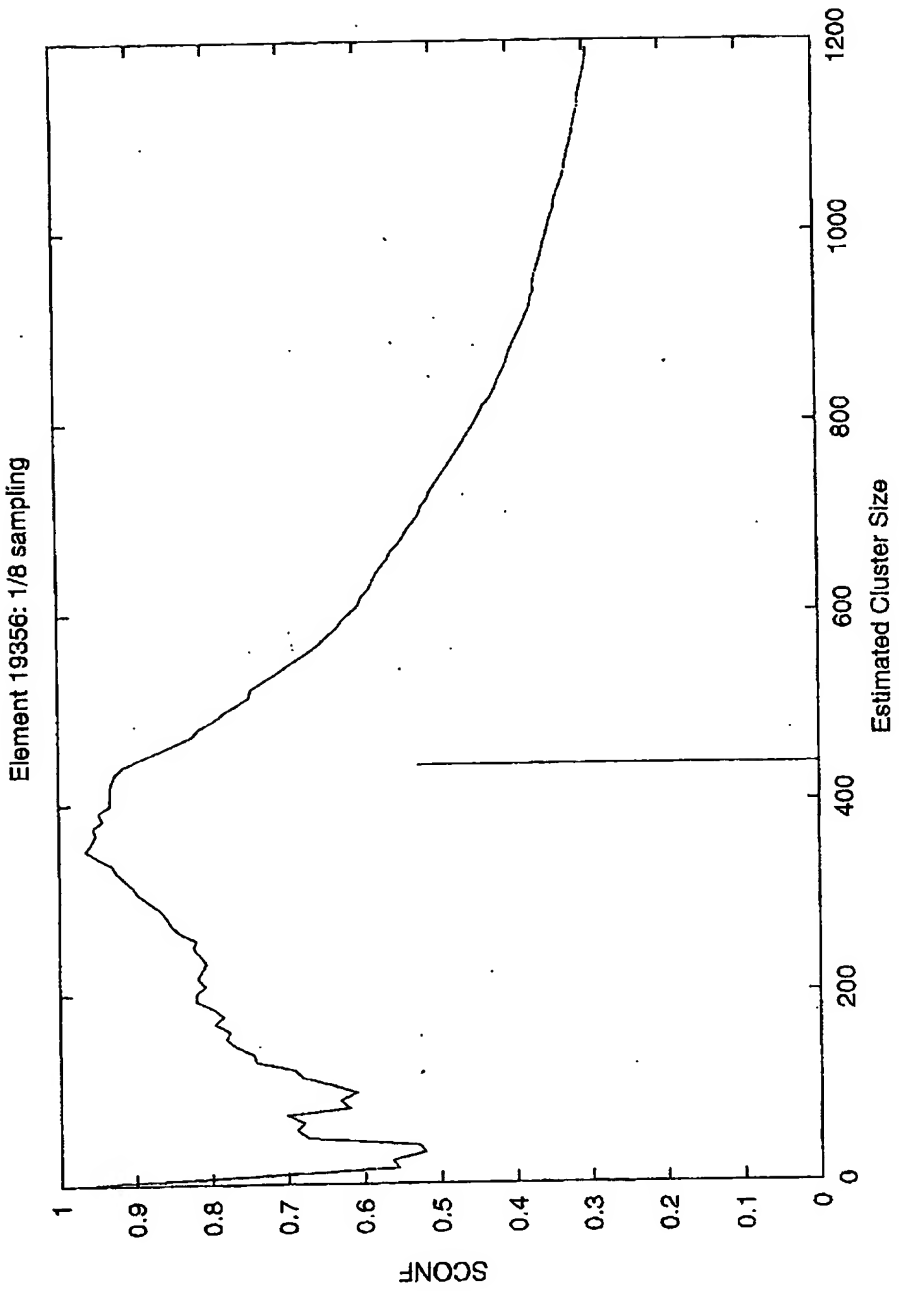


FIG 21